This application is submitted in the name of the following inventor(s):

Inventor	Citizenship	Residence City and State
Alan ROWE	United Kingdom	San Jose, California

The assignee is Network Appliance, Inc., a California corporation having an office at 495 East Java Drive, Sunnyvale, CA 94089.

Title of Invention

A Mechanism to Survive Server Failures When Using The CIFS Protocol

Background of the Invention

1. Field of the invention

This invention relates to transparent recovery of server failures and elective reboots while maintaining consistent data using the CIFS Filesystem protocol.

2. Related Art

The Common Internet File system (CIFS) protocol is defined by Microsoft. It enables collaboration on the internet by defining a remote file access protocol that allows applications to share data on local disks and network file servers. CIFS incorporates the same high-performance, multi-user read and write operations, locking, and file-sharing semantics that are the backbone of today's sophisticated enterprise

computer networks. With CIFS, users with different platforms and computers can share files without having to install new software.

CIFS generally runs over TCP/IP, and uses the SMB (Server Message Block) protocol found in Microsoft Windows® for file and printer access; therefore, CIFS will allow all PC applications, not just Web browsers, to open and share files across the Internet.

With CIFS, both the client and the server maintain state about filenames, file contents, directories, and various other aspects of the files and directories; thus CIFS is a "stateful" protocol. File content is cached via a cooperative process between client and server code, and this is where problems can occur. The state survives only as long as the session between the server and the client survives, and this session survives only as long as the underlying network connection (generally TCP/IP) survives.

When a server that is currently supporting one or more sessions fails or has to be purposefully rebooted, all sessions being supported are lost. CIFS has no protocol for re-establishment of a session after such a fatal error, or for synchronization of the client/server state to the pre-crash state. CIFS does support fault tolerance in the face of network and server failures where some CIFS clients can restore connections and reopen files that were open prior to interruption, however, any data that was currently being edited that had not been saved is lost. As a result, a server failure is regarded as a catastrophic event in the CIFS world.

Accordingly, it would be advantageous to provide a technique that addresses reestablishing server client sessions that were utilizing CIFS after a server

failure or elective reboot so that operation resumes where it ended prior to server unavailability.

Summary of the Invention

The invention includes a method and system for re-establishing sessions between a server and clients that were using the CIFS protocol. Two types of situations may occur. The first type occurs when a system administrator purposefully reboots the server or a purposeful takeover occurs in a clustered configuration. For these elective reboots of a server a series of tasks are performed; (1) the server stops accepting incoming CIFS requests, (2) the server completes processing of active CIFS requests, (3) all active CIFS state and networking state is captured in non-volatile storage (CIFS data structures are static at this point), (4) the server is rebooted, (5) state is rebuilt for the rebooted machine from that which was saved in non-volatile storage; in a takeover configuration state is made available through transmission or some form of non-uniform memory access, and (6) incoming CIFS requests are once again accepted and operation resumes.

The second type occurs when the server reboots without warning or there is an unplanned takeover due to server failure. These unplanned occurrences require the following tasks be performed; (1) state is saved persistently at predetermined intervals to non-volatile storage, (2) when the system crashes and reboots or is taken over, state is restored from the non-volatile storage, or in a takeover configuration, state is made available through transmission to a subsequent machine or through some form of non-uniform memory access, (3) operations that were in progress resume at the steps they

were at prior to the crash, (4) new CIFS requests are now accepted. All of the preceding is transparent to the clients and no data are lost.

Brief Description of the Drawings

Figure 1 illustrates a block diagram of a system for server failure survival when using the CIFS protocol.

Figure 2 illustrates a file server elective reboot/takeover process in a system for server failure survival when using the CIFS protocol.

Figure 3 illustrates a file server non-elective reboot/takeover process in a system for server failure survival when using the CIFS protocol.

Figure 4 illustrates a file server non-elective takeover process in a system to survive server failures when using the CIFS protocol.

Figure 5 illustrates critical state saving points in a mechanism to survive server failures when using the CIFS protocol.

Detailed Description of the Preferred Embodiment

In the following description, a preferred embodiment of the invention is described with regard to preferred process steps and data structures. Embodiment of the invention can be implemented using general purpose processors or special purpose

processors operating under program control, or other circuits, adapted to particular process steps and data structures described herein. Implementation of the process steps and data structures described herein would not require undue experimentation or further investigation.

Lexicography

The following terms refer to or relate to aspects of the invention as described below. The descriptions of general meanings of these terms are not intended to be limiting, only illustrative.

- client and server – These terms refer to a relationship between two devices, particularly to their relationship as a client and server, not necessarily to any particular physical devices.

- Client device and server device – These terms refer to devices taking on the role of a client device or a server device in a client-server relationship (such as an HTTP web client and web server). There is no particular requirement that any client devices or server devices be individual physical devices. They can each be a single device, a set of cooperating devices, a portion of a device, or some combination thereof.

- Procedure – A procedure is a self-consistent sequence of computerized steps that lead to a desired result. These steps are defined by one or more computer instructions. These steps are performed by a computer executing the instructions that define the steps. Thus, the term "procedure" can refer to a sequence of instructions, a sequence of instructions organized in a programmed-procedure or programmed-function, or a

sequence of instructions organized within programmed-processes executing in one or more computers.

- CIFS – Common Internet File System protocol defines a standard for remote file access using millions of computers at a time across different platforms that can share files.

- NetBIOS – An application programming interface (API) that augments the DOS BIOS by adding special functions for local-area networks (LANs).

- NBT – An implementation of Netbios over TCP/IP.

- SMB – Server Message Block. A message format used by DOS and Windows too share files, directories and devices. NetBIOS is based on the SMB format.

As noted above, these descriptions of general meanings of these terms are not intended to be limiting, only illustrative. Other and further applications of the invention, including extensions of these terms and concepts, would be clear to those of ordinary skill in the art after perusing this application. These other and further applications are part of the scope and spirit of the invention, and would be clear to those of ordinary skill in the art, without further invention or undue experimentation.

System Elements

Figure 1 shows a block diagram of a mechanism to survive server failures when using the CIFS protocol.

A preferred embodiment of the system 100 can include a client device 110, a client communication link 120, a communications network 130, a first server 140, a second server 150, a server communications link 160, an interconnect 170, and a mass storage 180.

The client device 110 includes a processor, memory, mass storage (not shown but understood by one skilled in the art). Typically, the client device 110 is associated with a user.

The client communication link 120 couples the client device 110 to the communications network 130. In a preferred embodiment, the communications network 130 includes an Internet, intranet, extranet, virtual private network, enterprise network, or another form of communication network.

The first server 140 includes a processor, a main memory (not shown but understood by one skilled in the art), and a first non-volatile storage 141. In a preferred embodiment the first server 130 and the client device 110 are separate devices, however, there is no requirement in any embodiment that they be separate devices. In a preferred embodiment, the first non-volatile storage 141 includes any electronic storage medium capable of retaining state without power or by some auxiliary power source (such as; non-volatile random access memory, magnetic and optical drives).

The second server 150 includes a processor, a main memory (not shown but understood by one skilled in the art), and a second non-volatile storage 151. In a preferred embodiment the second server 150 and the client device 110 are separate

devices, however, there is no requirement in any embodiment that they be separate devices. In a preferred embodiment, the first non-volatile storage 151 includes any electronic storage medium capable of retaining state without power or by some auxiliary power source (such as; non-volatile random access memory, magnetic and optical drives).

Additionally, the invention is applicable to both a standalone server and a server cluster; however, the second server 150 is used only in applications of the invention where the functions of the first server 130 are to be taken over by the second server 150. There is no requirement in any embodiment that the second server 150 be present in non-takeover applications of the invention.

A server communications link 160 couples the first server 140 and the second server 150 to the communication network 130.

An interconnect 170 couples the first server 140 to the second server 150 providing bi-directional communication between the two servers.

The mass storage 180 is coupled to both the first server 140 and the second server 150. In a preferred embodiment the mass storage 180 includes magnetic and optical disk arrays, and other devices capable of storing relatively large amounts of data.

Method of Operation – Elective Takeover and Elective Reboot

Figure 2 illustrates a file server elective reboot/takeover process, indicated by general reference character 200. The file server elective reboot/takeover process 200 initiates at a 'start' terminal 201. The file server elective reboot/takeover process 200 continues to a 'boot system' procedure 203 which enables the first server 140 to boot.

A "flag active' decision procedure 205 determines whether the first server 140 is rebooting following an elective reboot. If the 'flag active' decision procedure 205 determines that the first server 140 has been subjected to a reboot, the file server elective reboot/takeover process 200 continues to a 'restore state' procedure 227.

A 'receive CIFS requests' procedure 207 allows user requests to be received by the first server 140.

A 'process CIFS requests' procedure 209 allows the first server 140 to respond to requests from the client device 110 by providing access to data contained in the mass storage 180.

An 'initiate elective process?' procedure 211 determines whether the system is to be purposely taken offline (e.g. by the systems operator for maintenance purposes). If the 'initiate elective process?' procedure 211 determines that an elective shutdown has not been initiated, the file server elective reboot/takeover process 200 continues to the 'receive CIFS requests' procedure 207.

An 'ignore CIFS requests' procedure 213 causes the server device 140 to ignore all incoming CIFS requests from the client device 110. This is perceived by the client device 110 as a network delay and will not by itself terminate the session. The

client device 110 will resubmit CIFS requests until accepted or until the session is timed out (approximately 45 - 60 seconds from receipt of the first rejection by the client device 110) which ever comes first. The invention enables acceptance of CIFS requests prior to a session timing out.

A 'drain CIFS requests' procedure 215 ensures that all currently active CIFS requests are processed to completion.

An 'elective takeover?' decision procedure 217 determines whether an elective takeover has been selected by the systems operator. If the 'elective takeover' decision procedure 217 determines that an elective take over has been selected by the systems operator the file server elective reboot/ takeover process 200 continues to an 'elective takeover Save State' procedure' 223.

An 'elective reboot save state' procedure 219 causes the current state of the first server 140 to be stored in the first non-volatile storage 141. This includes the setting of the flag value to indicate a planned reboot of the first server 140.

A 'shut down system' procedure 221 causes the first server 140 to be shut down. The file server elective reboot/takeover process 200 terminates through an "end" terminal 229.

An 'elective takeover save state' procedure 223 causes the current state of the first server 140 to be stored in the first non-volatile storage 141 and the second non-volatile storage 151.

A 'takeover server restore state' procedure 225 allows the state of the first server 140 stored in the first non-volatile storage 141 to be transferred via the interconnect 170 and reconstituted on the second server device 150 or procured from the second non-volatile storage 151. At this point the second server 150 is supporting the sessions that were active on the first server 140 prior to elective takeover and CIFS processing within these sessions continues. The file server elective reboot/takeover process 200 terminates through an "end" terminal 229.

A 'restore state' procedure 227 allows the state of the first server 140 to be reconstituted to the state it was in prior to an elective reboot or non-elective reboot from the state stored in the first non-volatile storage 141. The file server elective reboot/takeover process 200 continues to a 'receive CIFS requests' procedure 207.

Method of Operation – Non-Elective Reboot.

Figure 3 illustrates a file server non-elective reboot process, indicated by general reference character 300. The file server non-elective reboot process 300 initiates at a 'start' terminal 301. The file server non-elective reboot process 300 continues to a 'boot system' procedure 303 which allows the first server 140 to boot.

A "flag active' decision procedure 305 determines whether an non-elective reboot has occurred. If the 'flag active' decision procedure 305 determines that the a non-elective reboot has occurred, the file server non-elective reboot process 300 continues to the 'resume normal operation' procedure 309.

A 'restore state' procedure 307 allows the first server 140 to reconstitute the state it was in prior to the non-elective reboot by copying state from that stored in the first non-volatile storage 141 or second non-volatile storage 151.

A 'resume normal operation' procedure 309 allows the first server 140 to once again accept and process CIFS requests and perform all functions it was executing prior to the non-elective reboot.

The file server non-elective reboot process 300 terminates through an "end" terminal 311.

Method of Operation – Non-Elective Takeover.

Figure 4 illustrates a file server non-elective takeover process, indicated by general reference character 4. The file server non-elective takeover process 400 initiates at a 'start' terminal 401. The file server non-elective takeover process 400 continues to a 'boot system' procedure 403 which allows the first server 140 to boot.

A 'receive CIFS requests' procedure 405 allows user requests to be received by the first server 140.

A 'process CIFS requests' procedure 407 allows the first server 140 to respond to requests from the client device 110 by providing access to data contained in the mass storage 180.

A 'save state' procedure 409 allows the state of the first server 140 to be saved to the first non-volatile storage 141 and the second non-volatile storage 151. In a preferred embodiment, reliably state saving in anticipation of a system failure may be performed at any one of a plurality of specific points within the processing of CIFS requests. For clarity in the description of this method of operation, 'save state' procedure 409 is indicated only once. The specific points for saving state are further discussed within this application.

A 'filer1 failure' decision procedure 411 determines whether the first server 140 has failed in some way. In a preferred embodiment, failure of the first server 140 would be detected by the second server 150. If the 'filer1 failure' decision procedure 411 determines that the first server 140 has not failed, the file server non-elective takeover process 400 continues to the 'receive CIFS requests' procedure 405.

A 'restore state' procedure 413 allows the state of the First server 140 prior to failure to be reconstitute on the second server 150 by copying state from that stored in the first non-volatile storage 141 or second non-volatile storage 151.

A 'filer2 takeover' procedure 415 completes the process by allowing the second filer 150 to resume processing of CIFS request where the first server 140 stopped.

The file server non-elective takeover process 400 terminates through an "end" terminal 417.

Method of Operation – Automatic State Saving

Figure 5 illustrates critical state saving points in a mechanism to survive server failures when using the CIFS protocol.

The saved state must always be in a consistent state. Automatic state saving must occur at specified points within a session of communication between the first server 140 and the client device 110 to ensure that the saved state is consistent.

POINT 1: State is saved prior to TCP acknowledging an incoming CIFS request. If the system fails prior to this, then the effect is as if the packet was never received, and retransmission by the client device 110 occurs. If the system fails after the acknowledgment is sent, then the system has a record that the request came in and it will be processed when state is restored.

POINT 2: State is saved prior to CIFS starting a SMB command. If the system fails prior to this, TCP will redeliver the TCP message to CIFS. If the system fails after this, the saved state indicates that the first server 140 started work on a CIFS operation. (Some single CIFS commands are composite operations: e.g. open, read, and close. In such cases, saving state is required before each component operation).

POINT 3: State is saved when a CIFS operation completes. If the system fails prior to this, the same CIFS operation is repeated creating the same result. If the system fails after this, the reply is sent again and TCP treats it as a duplicate.

POINT 4: State is saved after TCP acknowledges the reply. If the system fails prior to this, then the reply never happened and will be sent again. If the system fails after the acknowledgment but before the acknowledgment is saved, then we will duplicate the acknowledgment and normal TCP handling will process that without any problems. If the system fails after the save has occurred the acknowledgment will not be repeated.

These four points illustrate where state may be saved in a consistent manner, however, there are other points where state may be reliably saved and these points would be obvious to one skilled in the art.

Generality of the Invention

The invention has general applicability to various fields of use, not necessarily related to the services described above. For example, these fields of use can include one or more of, or some combination of, the following:

- In addition to general applicability to CIFS the invention has broad applicability to other transmission protocols.

Other and further applications of the invention, in its most general form, will be clear to those skilled in the art after perusal of this application, and are within the scope and spirit of the invention.

Alternate Embodiments

Although preferred embodiments are disclosed herein, many variations are possible which remain within the concept, scope, and spirit of the invention, and these variations would become clear to those skilled in the art after perusal of this application.